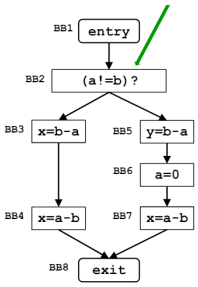
	Very Busy Expressions
Domain	Sets of Expressions
Direction	Backward: in[b] = f_b(out[b]) out[b] = ∧ in[succ(b)]
Transfer function	f_b(x) = Use_b ∪ (x - Kill_b) [1]
Meet Operation (∧)	∩
Boundary Condition	in[exit] = Ø
Initial interior points	in[b] = *U*

Dominio: {(a-b)(b-a)}

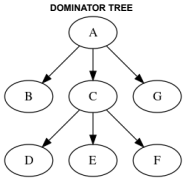
	Iterazione 1	
	IN[B]	**OUT[B]**
BB1 / Entry	/	(b-a)
BB2	Ø ∪ {(b-a) - Ø} = (b-a)	{(a-b)(b-a)} ∩ (b-a) = (b-a)
BB3	(b-a) ∪ {(a-b) - Ø} = {(a-b)(b-a)}	(a-b)
BB4	(a-b) ∪ Ø = (a-b)	Ø
BB5	(b-a) ∪ Ø = (b-a)	Ø
BB6	Ø ∪ {(a-b) - (a-b)} = Ø	(a-b)
BB7	(a-b) ∪ Ø = (a-b)	Ø
BB8 / Exit	Ø	/

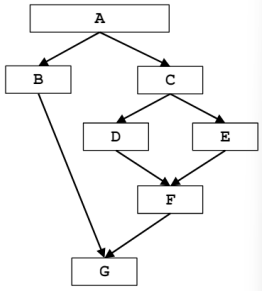


	Dominator Analysis
Domain	Sets of Basic Block
Direction	Forward: out[b] = f_b(in[b]) in[b] = ∧ out[pred(b)]
Transfer function	f_b(x) = Gen_b ∪ x [2]
Meet Operation (∧)	∩
Boundary Condition	out[entry] = {entry}
Initial interior points	out[b] = *U*

Dominio: {A, B, C, D, E, F, G}

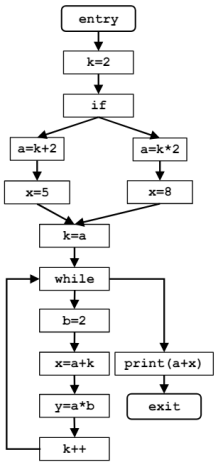
DOMINATOR TREE





	Iterazione 1	
	IN[B]	OUT[B]
A	Ø	A ∪ Ø = A
B	A	B ∪ A = (A, B)
C	A	C ∪ A = (A, C)
D	(A, C)	D ∪ (A, C) = (A, C, D)
E	(A, C)	E ∪ (A, C) = (A, C, E)
F	(A, C, D) ∩ (A, C, E) = (A, C)	F ∪ (A, C) = (A, C, F)
G	(A, B) ∩ (A, C, F) = A	G ∪ A = (A, G)

	Constant Propagation
Domain	Sets of Pair <variable, constant value>
Direction	Forward: out[b] = f_b(in[b]) in[b] = ∧ out[pred(b)]
Transfer function	f_b(x) = if all Use_b in x then Gen_b ∪ (x - Gen_b) else x - Gen_b [3]
Meet Operation (∧)	∩
Boundary Condition	out[entry] = Ø
Initial interior points	out[b] = *U*



	Iterazione 1	
	IN[B]	OUT[B]
k=2	Ø	<k, 2>
if 1		
a=k+2	<k, 2>	<a, 4> ∪ (<k, 2> - <a, *>) = <a, 4><k, 2>
x=5	<a, 4><k, 2>	<x, 5> ∪ (<a, 4><k, 2> - <x, *>) = <a, 4><k, 2><x, 5>
if 2		
a=k*2	<k, 2>	<a, 4> ∪ (<k, 2> - <a, *>) = <a, 4><k, 2>
x=8	<a, 4><k, 2>	<x, 8> ∪ (<a, 4><k, 2> - <x, *>) = <a, 4><k, 2><x, 8>
end if		
k=a	<a, 4><k, 2><x, 5> ∩ <a, 4><k, 2><x, 8> = <a, 4><k, 2>	<k, 4> ∪ (<a, 4><k, 2> - <k, *>) = <a, 4><k, 4>
while	out[k=k+1] ∩ <a, 4><k, 4> = <a, 4><k, 4>	<a, 4><k, 4>
b=2	<a, 4><k, 4>	<b, 2> ∪ (<a, 4><k, 4> - <b, *>) = <a, 4><b, 2><k, 4>
x=a+k	<a, 4><b, 2><k, 4>	<x, 8> ∪ (<a, 4><b, 2><k, 4> - <x, *>) = <a, 4><b, 2><k, 4><x, 8>
y=a*b	<a, 4><b, 2><k, 4><x, 8>	<y, 8> ∪ (<a, 4><b, 2><k, 4><x, 8> - <y, *>) = <a, 4><b, 2><k, 4><x, 8><y, 8>
k=k+1	<a, 4><b, 2><k, 4><x, 8><y, 8>	Ø ∪ (<a, 4><b, 2><k, 4><x, 8><y, 8> - <k, *>) = <a, 4><b, 2><x, 8><y, 8>

	Iterazione 2	
	IN[B]	OUT[B]
k=2	<a, 4><b, 2><x, 8><y, 8>	<a, 4><b, 2><k, 2><x, 8><y, 8>
if 1		
a=k+2	<a, 4><b, 2><k, 2><x, 8><y, 8>	<a, 4><b, 2><k, 2><x, 8><y, 8>
x=5	<a, 4><b, 2><k, 2><x, 8><y, 8>	<a, 4><b, 2><k, 2><x, 5><y, 8>
if 2		
a=k*2	<a, 4><b, 2><k, 2><x, 8><y, 8>	<a, 4><b, 2><k, 2><x, 8><y, 8>
x=8	<a, 4><b, 2><k, 2><x, 8><y, 8>	<a, 4><b, 2><k, 2><x, 8><y, 8>
end if		
k=a	<a, 4><b, 2><k, 2><x, 5><y, 8> ∩ <a, 4><b, 2><k, 2><x, 8><y, 8> = <a, 4><b, 2><k, 2><y, 8>	<a, 4><b, 2><k, 4><y, 8>
while	<a, 4><b, 2><x, 8><y, 8> ∩ <a, 4><b, 2><k, 4><y, 8> = <a, 4><b, 2><y, 8>	<a, 4><b, 2><y, 8>
b=2	<a, 4><b, 2><y, 8>	<a, 4><b, 2><y, 8>
x=a+k	<a, 4><b, 2><y, 8>	<a, 4><b, 2><y, 8>
y=a*b	<a, 4><b, 2><y, 8>	<a, 4><b, 2><y, 8>
k=k+1	<a, 4><b, 2><y, 8>	Ø ∪ (<a, 4><b, 2><y, 8> - <k, *>) = <a, 4><b, 2><y, 8>

	Iterazione 3	
	IN[B]	OUT[B]
k=2	<a, 4><b, 2><y, 8>	<a, 4><b, 2><k, 2><y, 8>
if 1		
a=k+2	<a, 4><b, 2><k, 2><y, 8>	<a, 4><b, 2><k, 2><y, 8>
x=5	<a, 4><b, 2><k, 2><y, 8>	<a, 4><b, 2><k, 2><x, 5><y, 8>
if 2		
a=k*2	<a, 4><b, 2><k, 2><y, 8>	<a, 4><b, 2><k, 2><y, 8>
x=8	<a, 4><b, 2><k, 2><y, 8>	<a, 4><b, 2><k, 2><x, 8><y, 8>
end if		
k=a	<a, 4><b, 2><k, 2><x, 5><y, 8> ∩ <a, 4><b, 2><k, 2><x, 8><y, 8> = <a, 4><b, 2><k, 2><y, 8>	<a, 4><b, 2><k, 2><y, 8>
while	<a, 4><b, 2><y, 8> ∩ <a, 4><b, 2><k, 2><y, 8> = <a, 4><b, 2><y, 8>	<a, 4><b, 2><y, 8>
b=2	<a, 4><b, 2><y, 8>	<a, 4><b, 2><y, 8>
x=a+k	<a, 4><b, 2><y, 8>	<a, 4><b, 2><y, 8>
y=a*b	<a, 4><b, 2><y, 8>	<a, 4><b, 2><y, 8>
k=k+1	<a, 4><b, 2><y, 8>	Ø ∪ (<a, 4><b, 2><y, 8> - <k, *>) = <a, 4><b, 2><y, 8>

	Iterazione 4	
	IN[B]	OUT[B]
k=2	<a, 4><b, 2><y, 8>	<a, 4><b, 2><k, 2><y, 8>
a=k+2	<a, 4><b, 2><k, 2><y, 8>	<a, 4><b, 2><k, 2><y, 8>
a=k*2	<a, 4><b, 2><k, 2><y, 8>	<a, 4><b, 2><k, 2><y, 8>

[1] Use_b = Le operazioni che vengono utilizzate nel blocco

Kill_b = Le espressioni definite successivamente alle quali almeno uno degli operandi viene ridefinito nel blocco attuale

[2] Gen_b = blocco attuale

[3] Permette di verificare se la variabile generata è ottenuta da sole costanti e quindi la aggiunge all'insieme delle coppie costanti, altrimenti la rimuove dall'insieme